椭圆函数的精细积分算法

钟万勰 [1)]姚 征
(大连理工大学工业装备结构分析国家重点实验室, 大连 116023)

摘 要 椭圆函数是一种特殊的双周期复变函数, 广泛应用于工程问题中, 尤其非线性问题中居多. 在工程中遇到的椭圆函数以二阶椭圆函数为主, 而且很多复杂的椭圆函数都可以通过变换由二阶椭圆函数得到. 二阶椭圆函数包括 Jacobi 椭圆函数和 Weierstrass 椭圆函数. 它们都可以进行幂级数展开, 直接计算很不方便. 椭圆函数的重要性质之一就是具有加法定理, 所以可以利用精细积分法 [1] 求解. 通过使用精细积分法成功求解了二阶椭圆函数, 并指出精细积分法是一种求解椭圆函数的高效稳定的方法.

关键词 *精细积分*, Jacobi *椭圆函数*, Weierstrass *椭圆函数*, *双周期*

引 言

精细积分法首先解决了时不变系统的时间积分问题, 并取得了达到计算机精度的高精度结果. 精细积分法有 2 个要点: (1) 运用 2^N 类算法 [1,3,4]; (2) 把注意力放在增量上, 而不是全量. 解决时不变系统之后, 各种近似方法被广泛的应用于求解其他问题, 例如时变系统或非线性系统的时间积分等.

在工程问题中常遇到椭圆函数数值求解的问题, 该问题可以通过查表和使用软件来解决, 例如 Matlab 和 IMSL 等软件就有求解椭圆函数的模块. 本文利用精细积分法实现了对 Jacobi 椭圆函数和 Weierstrass 椭圆函数的数值求解, 大量的数值算例指出精细积分法得出的结果具有极高的精度, 而且无论是在效率上还是在稳定性上精细积分法都明显的优于 IMSL 数学库的椭圆函数模块.

1 Jacobi 椭圆函数的精细积分算法

第二类 Legendre 椭圆积分可以定义为

$$u = \int_0^Z \frac{\mathrm{d}z}{\sqrt{(1-z^2)(1-k^2z^2)}} \tag{1}$$

Jacobi 椭圆函数 $sn\,u = z$ 定义为第二类 Legendre 椭圆积分的反函数. 另外两个基本 Jacobi 椭圆函数分别定义为

$$cn\,u = \sqrt{1-sn^2u} = \sqrt{1-z^2}, \quad dn\,u = \sqrt{1-k^2sn^2u} = \sqrt{1-k^2z^2} \tag{2}$$

其中, 参数 k 称为模数. 这三个 Jacobi 椭圆函数的幂级数展开式分别为 [2]

1) Email: wxzhong@dlut.edu.cn, wind0411@student.dlut.edu.cn

© 1994-2013 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

$$\left.\begin{aligned} sn\,u &= u - \frac{1}{3!}(1+k^2)u^3 + \frac{1}{5!}(1+14k^2+k^4)u^5 - \cdots \\ cn\,u &= 1 - \frac{1}{2!}u^2 + \frac{1}{4!}(1+4k^2)u^4 - \frac{1}{6!}(1+44k^2+16k^4)u^6 + \cdots \\ dn\,u &= 1 - \frac{1}{2!}k^2u^2 + \frac{1}{4!}(4k^2+k^4)u^4 - \frac{1}{6!}(16k^2+44k^4+k^6)u^6 + \cdots \end{aligned}\right\} \tag{3}$$

相应的加法公式为

$$sn(2v) = \frac{2snv\cdot cnv\cdot dnv}{1-k^2\cdot sn^4v},\quad cn(2v) = \frac{cn^2v - sn^2v\cdot dn^2v}{1-k^2\cdot sn^4v},\quad dn(2v) = \frac{dn^2v - kn^2v\cdot sn^2v\cdot cn^2v}{1-k^2\cdot sn^4v} \tag{4}$$

现在需要解决的问题是：给出坐标变量 u 和模数 k 数值求解这三个 Jacobi 椭圆函数. 利用这些函数的加法定理 (4), 精细积分法可以用于求解该类问题.

为了进行数值积分，引入一个小的空间步长，记为： η. 于是产生一系列等步长的空间间隔为

$$u_0 = 0,\quad u_1 = \eta,\quad u_k = k\eta,\quad \cdots,\quad u_{2^L} = 2^L\eta = u \tag{5}$$

假设这三个 Jacobi 椭圆函数位于 η 点的值都已经求得，那么通过利用加法定理 (4) 就可方便的求出函数位于 u 点的值. 因此，原问题变为在 η 点对 Jacobi 椭圆函数进行数值求解. 令

$$\tau = \eta/m,\ \text{其中}\ m = 2^N,\ \text{例如}\ N = 20,\quad m = 1048576 \tag{6}$$

由于 η 本来是不大的空间区段，则 $\tau = \eta/m$ 将是非常小的一个空间区段了. 因此对于空间区段 τ, 有

$$\left.\begin{aligned} sn\tau &\approx \tau - \frac{1}{3!}(1+k^2)\tau^3 + \frac{1}{5!}(1+14k^2+k^4)\tau^5 = S_1 \\ cn\tau &\approx 1 - \frac{1}{2!}\tau^2 + \frac{1}{4!}(1+4k^2)\tau^4 - \frac{1}{6!}(1+44k^2+16k^4)\tau^6 = 1 + C_1 \\ dn\tau &\approx 1 - \frac{1}{2!}k^2\tau^2 + \frac{1}{4!}(4k^2+k^4)\tau^4 - \frac{1}{6!}(16k^2+44k^4+k^6)\tau^6 = 1 + D_1 \end{aligned}\right\} \tag{7}$$

因 τ 很小，幂级数 4 项展开应以足够. 在上式中， S_1, C_1 和 D_1 相对于单位 1 是非常小的量. 因此在计算过程中至关重要的一点是数值的存储只能是 (7) 式中的 S_1, C_1 和 D_1, 而不是 $1+C_1$ 和 $1+D_1$. 因为 C_1 和 D_1 很小，当它们与单位 1 相加时，就会成为尾数，在计算机的舍入操作中，其精度将丧失殆尽. S_1, C_1 和 D_1 就是增量 [1,4]. 这就是以上提到的精细积分法的第二个要点.

把 (7) 式代入 (4) 式，并令 $snv = S_i$ 可以得到

$$\left.\begin{aligned} sn(2v &= F_S(S_i) = \frac{2S_i(1+C_i)(1+D_i)}{1-k^2S_i^4} = S_{2i} \\ cn(2v) &= 1 + F_C(C_i) = 1 + \frac{k^2S_i^4 + 2C_i + C_i^2 - S_i^2(1+D_i)^2}{1-k^2S_i^4} = 1 + C_{2i} \\ dn(2v) &= 1 + F_D(D_i) = 1 + \frac{k^2S_i^4 + 2D_i + D_i^2 - k^2S_i^2(1+C_i)^2}{1-k^2S_i^4} = 1 + D_{2i} \end{aligned}\right\} \tag{8}$$

© 1994-2013 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

因此，通过 (8) 式我们可以得到 S_m, C_m 和 D_m，从而求得 $sn(\eta), cn(\eta)$ 和 $dn(\eta)$. 这样的计算一共需要执行 N 次，而且只有 S_i, C_i 和 D_i $(i = 2^0, 2^1, \ldots, 2^N)$ 被计算并存入内存. (8) 式的 N 次计算相当于一下语句

$$\text{for (iter = 0; iter < N; iter ++)}\ i = 2^{\text{iter}};\quad S_{2i} = F_S(S_i);\quad C_{2i} = F_C(C_i);\quad D_{2i} = F_D(D_i) \tag{9}$$

当以上语句的循环结束后，再执行

$$sn(m\tau) = sn(\eta) = S_m,\quad cn(m\tau) = cn(\eta) = 1 + C_m,\quad dn(m\tau) = dn(\eta) = 1 + D_m \tag{10}$$

便可. 由于 N 次叠代后，S_m, C_m 和 D_m 已不再是很小的量了，这个加法已没有严重的舍入误差了 [1,4]. 以上便是 Jacobi 椭圆函数的精细积分算法.

2 Weierstrass 椭圆函数的精细积分算法

第一类 Weierstrass 椭圆积分定义为

$$u = \int_{\infty}^{z} \frac{\mathrm{d}z}{\sqrt{4z^3 - g_2 z - g_3}} \tag{11}$$

Weierstrass 椭圆函数 $z = \wp(u)$ 定义为第一类 Weierstrass 椭圆积分的反函数. 上式中的 g_2，g_3 被称为 Weierstrass 椭圆函数的不变量，它们是 Weierstrass 椭圆函数的 2 个周期 ω 和 ω' 的函数

$$g_2 = 60S_4 = \sum_{m,m'}{}' \frac{140}{w^6},\quad g_3 = 140S_6 = \sum_{m,m'}{}' \frac{140}{w^6} \tag{12}$$

其中：$w = 2m\omega + 2m'\omega'$; 式 (12) 中的 m 和 m' 是对全体整数求和，但 m 和 m' 不能同时为 0.

Weierstrass 椭圆函数的精细积分算法与第一节中得到的 Jacobi 椭圆函数的精细积分算法相类似. 因此，这里仅阐述该算法的主要过程与公式.

Weierstrass 椭圆函数的幂级数展开式和加法定理分别为 [2]

$$\wp(u) = \frac{1}{u^2} + \frac{g_2}{20}u^2 + \frac{g_3}{28}u^4 + \frac{g_2^2}{1\,200}u^6 + \frac{g_2 g_3}{560}u^8 + \cdots \tag{13}$$

$$\wp(2v) = \frac{\wp^4(v) + \frac{1}{2}g_2\wp^2(v) + 2g_3\wp(v) + \frac{1}{16}g_2^2}{4\wp^3(v) - g_2\wp(v) - g_3} \tag{14}$$

给出一个极小的空间区段：$\tau = \eta/m$, 其中 $m = 2^N$, 并令

$$P_1 = \frac{g_2}{20}\tau^2 + \frac{g_3}{28}\tau^4 + \frac{g_2^2}{1\,200}\tau^6 + \frac{g_2 g_3}{560}\tau^8 \tag{15}$$

于是，方程 (13) 可以重写为

$$\wp(\tau) \approx \frac{1}{\tau^2} + P_1 \tag{16}$$

把 (16) 式代入 (14) 式，并令：$\wp(v) = \dfrac{1}{v^2} + P_i$, $v = 2^i\tau$, 加法定理 (14) 可以重写为

$$\wp(2v) = \frac{1}{(2v)^2} + \frac{1}{4}P_i + \frac{\frac{3}{4}g_2(v + v^3P_i)^2 + \frac{9}{4}g_3(v^4 + v^6P_i) + \frac{1}{16}g_2^2v^6}{4(1 + v^2P_i)^3 - g_2(v^4 + v^6P_i) - v^6g_3} =$$

© 1994-2013 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

$$\frac{1}{(2v)^2}+F(v,P_i)=\frac{1}{(2v)^2}+P_{2i} \tag{17}$$

在式 (17) 中 P_{2i} 是常规部分. Weierstrass 椭圆函数的精细积分算法可以按下面过程实现

首先，按照以下语句进行 N 次循环计算

$$\text{for (iter} = 0;\ \text{iter} < N;\ \text{iter} + +\,)\ i = 2^{\text{iter}},\quad P_{2i}=F(i\tau,P_i) \tag{18}$$

注意 只有常规项 P_{2i} 的值而不是 $\wp(i\tau)$ 的值存入计算机内存.

然后，当进行完以上的循环计算时， Weierstrass 椭圆函数的数值结果可以由下式得到

$$\wp(u)=\wp(m\tau)=P_m+\frac{1}{u^2} \tag{19}$$

因此，通过以上推导我们得到了 Weierstrass 椭圆函数的精细积分算法.

3 数值算例与误差分析

在本节中将讨论精细积分算法的精度, 效率和稳定性, 相应的误差分析也将给出. 在本节中精细积分算法的程序采用 Fortran90 语言编写, 而且在全部的算例中如无特殊说明都取 $N=20$.

3.1 Jacobi 椭圆函数

(A) 蜕化为三角函数演算

当 Jacobi 椭圆函数的模数 $k=0$ 时， Jacobi 椭圆函数就蜕化为三角函数

$$sn\,u|_{k=0}=\sin u,\quad cn\,u|_{k=0}=\cos u,\quad dn\,u|_{k=0}=\tan u \tag{20}$$

三角函数的精确数值求解早已在各软件和计算机语言中实现，通过和相应的三角函数值对比，就可以得出精细积分法相对精度.

在复域内取 500 000 个点 (不包括极点), 用精细积分法计算 Jacobi 椭圆函数当 $k=0$ 时在各点的值，并用各点相应的三角函数值对比. 结果给出全部点的相对误差均小于 10^{-15}.

(B) 特殊点验算

Jacobi 椭圆函数在每个单胞内都存在特殊点

$$\left.\begin{aligned}&sn\,K=1,\quad cn\,K=0,\quad dn\,K=k'\\&sn(K+iK')=\frac{1}{k},\quad cn(K+iK')=-\frac{ik'}{k},\quad dn(K+iK')=0\\&sn(iK')=\infty,\quad cn(iK')=\infty,\quad dn(iK')=\infty\end{aligned}\right\} \tag{21}$$

上式中的 k' 称作补模数. K 和 K' 为分别与 k 和 k' 相对应的全椭圆积分，相关的定义可以参看文 [2].

通过对取不同模数 k 的 500 000 点的验算，发现相对误差的变化范围为 $10^{-6}\sim10^{-16}$. 分析相对误差产生的原因发现是由于：当 k 和 k' 接近 0 或 1 时， K 和 K' 无法得到精度高的近似解.

(C) 恒等式验算

Jacobi 椭圆函数存在如下恒等式

$$sn^2u+cn^2u=1,\quad dn^2u+k^2sn^2u=1 \tag{22}$$

© 1994-2013 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

对于不同模数 k 的 Jacobi 椭圆函数在复空间内的任意一点 (不包括极点) 都应该满足 (22) 式. 在一个单胞内取 500 000 个不同的坐标点作为算例，计算 Jacobi 椭圆函数的值并回代 (22) 式得到：所有算例的相对误差均小于 10^{-14}.

3.2 Weierstrass 椭圆函数

Weierstrass 椭圆函数没有类似于 Jacobi 椭圆函数的恒等式；而且对于给定的 g_2, g_3 的情况无法求得该函数的周期 2ω 和 $2\omega'$. 所以直接的数值检验方法较少. Weierstrass 椭圆函数与 Jacobi 椭圆函数之间存在如下关系

$$sn^2 v = \frac{e_1 - e_2}{\wp\left(\dfrac{v}{\sqrt{e_1 - e_2}}\right) - e_2}, \quad k^2 = \frac{e_3 - e_2}{e_1 - e_2} \tag{23}$$

式中的 e_1, e_2 和 e_3 称作 Weierstrass 椭圆函数的平稳值，同时也是下面三次方程得根

$$4z^3 - g_2 z - g_3 = 0 \tag{24}$$

由于 Jacobi 椭圆函数已经可以精确求解，故可以利用式 (23), 式 (24) 式将求得的 Weierstrass 椭圆函数与相应的 Jacobi 椭圆函数对比, 进而验证 Weierstrass 椭圆函数的精度. 同样计算 500 000 个算例，得到：所有算例的相对误差均小于 10^{-14}.

对于给定周期 2ω 和 $2\omega'$ 的情况 (采用的精细积算法与第 2 节中所论述的不同，由于篇幅关系就不在这里详细论述，具体算法将在另一篇文章中给出), 就可以进行周期性验证. 在复空间内取一对坐标 (x_1, y_1) 和 (x_2, y_2), 这两个坐标点的空间间隔为

$$(x_2 + y_2 i) - (x_1 + y_1 i) = l_1 \cdot 2\omega + l_2 \cdot 2\omega' \tag{25}$$

上式中 l_1 和 l_2 为整数，且 $|l_1| + |l_2| = 10$. 根据椭圆函数的双周期性，函数应该在这两点取相同的函数值. 分别计算两点的值并对比就可以得到相应的相对误差.

验算 500 000 对坐标点，得到相对误差均小于 10^{-12}.

3.3 与 IMSL 计算模块对比

现在很多软件都存在计算 Jacobi 椭圆函数的模块，例如 Matlab,Mathematica 和 IMSL. 这里我们将 Jacobi 椭圆函数的精细积分法程序与 IMSL fortran 库的 Jacobi 函数模块在进行一下对比.

3.3.1 计算速度对比

分别用 IMSL 和精细积分法将 200 000 个算例计算 3 次，并将每种方法所消耗的时间计入表 1.

表 1 IMSL 和精细积分的效率对比

	消耗的时间 (s)		
	IMSL	精细积分法	
		$N = 10$	$N = 20$
第 1 次	4.13600000000000	1.68300000000000	3.05400000000000
第 2 次	4.05500000000000	1.75200000000000	3.09400000000000
第 3 次	4.11600000000000	1.76300000000000	3.05500000000000

当 $N = 20$ 时， IMSL 和精细积分法的精度都可以达到双精度的要求. 但从表 1 中可以看出，精细积分法的效率大约是 IMSL 的 1.3 倍. 而且调用精细积分法求解时，一次计算可以同

© 1994-2013 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

时给出 sn, cn, dn 的值. 这就意味着如果一个包含椭圆函数的表达式中同时需要求解同一点的不同 Jacobi 椭圆函数值那么精细积分法的效率就可以在提高 2 倍 (同时计算 2 种 Jacobi 椭圆函数) 或 3 倍 (同时计算 3 种 Jacobi 椭圆函数), 此时精细积分法的效率就是 IMSL 的 2.6 或 3.9 倍了, 这是非常可观的, 如果该问题需要大量迭代则可以节约很多时间.

3.3.2 稳定性对比

Jacobi 椭圆函数在每个单胞内存在 2 个奇点, 在靠近奇点的区域是无论是 IMSL 模块还是精细积分法都会产生误差. 如果在复空间划分一个区域: $(-10-10i, 10-10i, 10+10i, -10+10i)$, 相对模数 $k^2 = 0.5$, 在该区域内 3 种 Jacobi 椭圆函数都存在完整的单胞. 在该区域内沿实轴等间距分别作 $n-1$ 条虚轴的平行线, 同理沿虚轴也作 $n-1$ 条实轴的平行线. 空间被刨分出 $(n+1)\times(n+1)$ 个点, 这些点都应满足恒等式 (22), 可以分别计算出每个点的值代入 (22) 式所得的相对误差. 随着 n 的增加, 近奇点区域内的点也会增加, 相对误差大的点也会增多. 在表 2 中列出了相对误差的统计结果:

表 2 IMSL 与精细积分的稳定性对比

点的个数	相对误差 $> 10^{-12}$ 的点的个数		最大相对误差	
$n\times n$	IMSL	精细积分	IMSL	精细积分
40401	8104	0	$1.2910277291\times10^{-9}$	$4.8233240330\times10^{-13}$
161604	32380	37	$7.7852746471\times10^{-9}$	$1.4551915228\times10^{-11}$
1010025	201709	228	$2.3911707318\times10^{-7}$	$1.1641532269\times10^{-10}$

从表 2 不难看出, 精细积分法无论是精度还是稳定性都明显的优于 IMSL 模块.

4 结论与展望

本文利用精细积分法对二阶椭圆函数进行了成功的求解, 并通过大量的算例说明精细积分法具有很高的精度与稳定性. 但目前还有一些工作需要进一步完成:

(1) Weierstrass 椭圆函数还需要作更多的误差分析和算法对比;

(2) 将精细积分法应用于其它特殊函数的求解.

参 考 文 献

1 Wanxie Zhong. On precise integration method. *Journal of Computational and Applied Mathematics*, 2004, 163

2 高本庆. 椭圆函数及其应用. 北京: 国防工业出版社, 1991

3 钟万勰. 结构动力方程的精细时程积分法. 大连理工大学学报, 1994, 43(2): 1865~1872

4 钟万勰. 应用力学对偶体系. 北京: 科学出版社, 2002

© 1994-2013 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net